The Complexity of Counting Models of Linear-time Temporal Logic

Joint work with Hazem Torfah

Martin Zimmermann

Saarland University

September 4th, 2014

Highlights 2014, Paris, France

Counting Complexity

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathrm{P}$ if there is an NP machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Counting Complexity

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathrm{P}$ if there is an NP machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

For complexity class $\mathcal{C}$:

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathcal{C}$ if there is an NP machine $\mathcal{M}$ with oracle in $\mathcal{C}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Counting Complexity

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathrm{P}$ if there is an NP machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

For complexity class $\mathcal{C}$:

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathcal{C}$ if there is an NP machine $\mathcal{M}$ with oracle in $\mathcal{C}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Remark: $f \in \#\mathcal{C}$ implies $f(w) \in \mathcal{O}(2^{p(|w|)})$ for some polynomial p.

Counting Complexity

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathrm{P}$ if there is an NP machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

For complexity class $\mathcal{C}$:

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathcal{C}$ if there is an NP machine $\mathcal{M}$ with oracle in $\mathcal{C}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Remark: $f \in \#\mathcal{C}$ implies $f(w) \in \mathcal{O}(2^{p(|w|)})$ for some polynomial p.

We need *larger* counting classes.

- $f : \Sigma^* \to \mathbb{N}$ is in $\#_d\mathrm{PSPACE}$, if there is a nondeterministic polynomial-space Turing machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Counting Complexity

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathrm{P}$ if there is an NP machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

For complexity class $\mathcal{C}$:

- $f : \Sigma^* \to \mathbb{N}$ is in $\#\mathcal{C}$ if there is an NP machine $\mathcal{M}$ with oracle in $\mathcal{C}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Remark: $f \in \#\mathcal{C}$ implies $f(w) \in \mathcal{O}(2^{p(|w|)})$ for some polynomial p.

We need *larger* counting classes.

- $f : \Sigma^* \to \mathbb{N}$ is in $\#_d\textsc{Pspace}$, if there is a nondeterministic polynomial-space Turing machine $\mathcal{M}$ such that $f(w)$ is equal to the number of accepting runs of $\mathcal{M}$ on w.
- Analogously: $\#_d\textsc{Exptime}$, $\#_d\textsc{Expspace}$, and $\#_d2\textsc{Exptime}$.

Counting Complexity

Lemma

$\#\mathrm{P}$

Counting Complexity

Lemma

$\#\mathrm{P} \subseteq \#\textsc{Pspace}$

Counting Complexity

Lemma

$\#\mathrm{P} \subseteq \#\textsc{Pspace} \subseteq \#\textsc{Exptime}$

Counting Complexity

Lemma

$\#\mathrm{P} \subseteq \#\textsc{Pspace} \subseteq \#\textsc{Exptime} \subseteq \#\mathrm{N}\textsc{Exptime}$

Counting Complexity

Lemma

$\#\mathrm{P} \subseteq \#\textsc{Pspace} \subseteq \#\textsc{Exptime} \subseteq \#\textsc{NExptime} \subseteq \#\textsc{Expspace}$

Counting Complexity

Lemma

$\#\mathrm{P} \subseteq \#\textsc{Pspace} \subseteq \#\textsc{Exptime} \subseteq \#\textsc{NExptime} \subseteq \#\textsc{Expspace} \subseteq \#2\textsc{Exptime}$

Counting Complexity

Lemma

$\#_d$Pspace

#P ⊆ #Pspace ⊆ #Exptime ⊆ #NExptime ⊆ #Expspace ⊆ #2Exptime

Counting Complexity

Lemma

$\#_d\text{PSPACE} \subseteq \#_d\text{EXPTIME}$

$\#\text{P} \subseteq \#\text{PSPACE} \subseteq \#\text{EXPTIME} \subseteq \#\text{NEXPTIME} \subseteq \#\text{EXPSPACE} \subseteq \#2\text{EXPTIME}$

Counting Complexity

Lemma

$$\#_d\textsc{Pspace} \subseteq \#_d\textsc{Exptime} \quad \subsetneq \quad \#_d\textsc{Expspace}$$

$$\#\mathrm{P} \subseteq \#\textsc{Pspace} \subseteq \#\textsc{Exptime} \subseteq \#\mathrm{N}\textsc{Exptime} \subseteq \#\textsc{Expspace} \subseteq \#2\textsc{Exptime}$$

Counting Complexity

Lemma

$$\#_d\textsc{Pspace} \subseteq \#_d\textsc{Exptime} \quad \subsetneq \quad \#_d\textsc{Expspace} \subseteq \#_d 2\textsc{Exptime}$$

$$\#\text{P} \subseteq \#\textsc{Pspace} \subseteq \#\textsc{Exptime} \subseteq \#\text{N}\textsc{Exptime} \subseteq \#\textsc{Expspace} \subseteq \#2\textsc{Exptime}$$

Counting Complexity

Lemma

$$
\begin{array}{ccccccccccc}
 & & \#_d\textsc{Pspace} & \subseteq & \#_d\textsc{Exptime} & & \subsetneq & & \#_d\textsc{Expspace} & \subseteq & \#_d2\textsc{Exptime} \\
 & & \rotatebox{90}{$\subsetneq$} & & \rotatebox{90}{$\subsetneq$} & & & & \rotatebox{90}{$\subsetneq$} & & \rotatebox{90}{$\subsetneq$} \\
\#\mathrm{P} & \subseteq & \#\textsc{Pspace} & \subseteq & \#\textsc{Exptime} & \subseteq & \#\mathrm{N}\textsc{Exptime} & \subseteq & \#\textsc{Expspace} & \subseteq & \#2\textsc{Exptime}
\end{array}
$$

Counting Complexity

Lemma

$$
\begin{array}{ccccccccccc}
 & & \#_d\textsc{Pspace} & \subseteq & \#_d\textsc{Exptime} & & \subsetneq & & \#_d\textsc{Expspace} & \subseteq & \#_d2\textsc{Exptime} \\
 & & \rotatebox{-90}{$\subsetneq$} & & \rotatebox{-90}{$\subsetneq$} & & & & \rotatebox{-90}{$\subsetneq$} & & \rotatebox{-90}{$\subsetneq$} \\
\#\mathrm{P} & \subseteq & \#\textsc{Pspace} & \subseteq & \#\textsc{Exptime} & \subseteq & \#\mathrm{N}\textsc{Exptime} & \subseteq & \#\textsc{Expspace} & \subseteq & \#2\textsc{Exptime}
\end{array}
$$

Reductions:

- f is $\#\mathrm{P}$-hard, if there is a polynomial time computable function r s. t. $f(r(\mathcal{M}, w))$ is equal to the number of accepting runs of $\mathcal{M}$ on w.

Counting Complexity

Lemma

$$
\begin{array}{ccccccccccc}
 & & \#_d\textsc{Pspace} & \subseteq & \#_d\textsc{Exptime} & & \subsetneq & & \#_d\textsc{Expspace} & \subseteq & \#_d2\textsc{Exptime} \\
 & & \rotatebox{90}{$\subsetneq$} & & \rotatebox{90}{$\subsetneq$} & & & & \rotatebox{90}{$\subsetneq$} & & \rotatebox{90}{$\subsetneq$} \\
\#\mathrm{P} & \subseteq & \#\textsc{Pspace} & \subseteq & \#\textsc{Exptime} & \subseteq & \#\textsc{NExptime} & \subseteq & \#\textsc{Expspace} & \subseteq & \#2\textsc{Exptime}
\end{array}
$$

Reductions:

- f is $\#\mathrm{P}$-hard, if there is a polynomial time computable function r s. t. $f(r(\mathcal{M}, w))$ is equal to the number of accepting runs of $\mathcal{M}$ on w.
- Hardness for other classes analogously.
- Completeness as usual.

Counting Word-Models

Theorem

- *The following problem is $\#\mathrm{P}$-complete: Given an LTL formula φ and a bound k (in unary), how many k-word-models does φ have?*

Counting Word-Models

Theorem

- *The following problem is $\#\mathrm{P}$-complete: Given an LTL formula φ and a bound k (in unary), how many k-word-models does φ have?*
- *The following problem is $\#_d\textsc{Pspace}$-complete: Given an LTL formula φ and a bound k (in binary), how many k-word-models does φ have?*

Counting Word-Models

Theorem

- *The following problem is $\#\mathrm{P}$-complete: Given an LTL formula φ and a bound k (in unary), how many k-word-models does φ have?*
- *The following problem is $\#_d\textsc{Pspace}$-complete: Given an LTL formula φ and a bound k (in binary), how many k-word-models does φ have?*

Lower bound: $\textsc{Pspace}$-hardness of LTL satisfiability **[SC85]** made one-to-one

Counting Word-Models

Theorem

- *The following problem is $\#\mathrm{P}$-complete: Given an LTL formula φ and a bound k (in unary), how many k-word-models does φ have?*
- *The following problem is $\#_d\textsc{Pspace}$-complete: Given an LTL formula φ and a bound k (in binary), how many k-word-models does φ have?*

Lower bound: $\textsc{Pspace}$-hardness of LTL satisfiability **[SC85]** made one-to-one

Upper bound: Guess word of length k and model-check it

Counting Tree-Models with Unary Bounds

Theorem

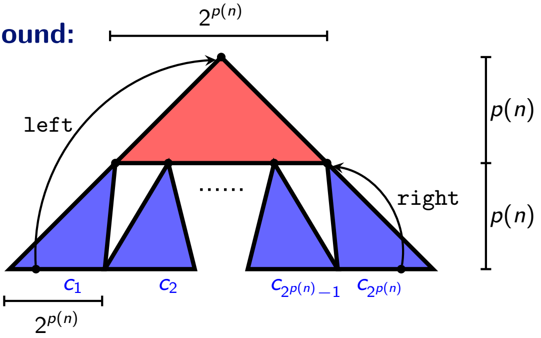
The following problem is $\#_d$EXPTIME-complete: Given an LTL formula φ and a bound k (in unary), how many k-tree-models does φ have?

Counting Tree-Models with Unary Bounds

Theorem

The following problem is $\#_d$EXPTIME-complete: Given an LTL formula φ and a bound k (in unary), how many k-tree-models does φ have?

- **Lower bound:**

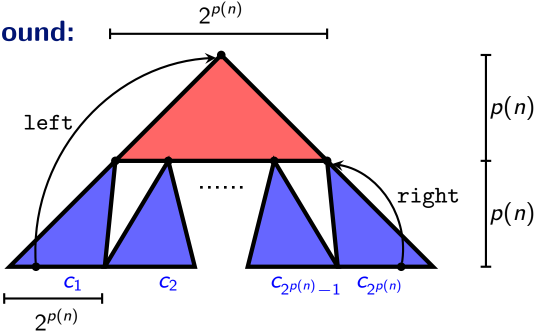


Counting Tree-Models with Unary Bounds

Theorem

The following problem is $\#_d$EXPTIME-complete: Given an LTL formula φ and a bound k (in unary), how many k-tree-models does φ have?

- **Lower bound:**



- **Upper bound:** Guess tree of height k and model-check it.

Counting Tree-Models with Binary Bounds

Theorem

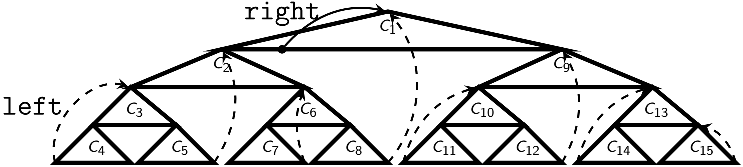
The following problem is $\#_d$EXPSPACE-hard and in $\#_d$2EXPTIME: Given an LTL formula φ and a bound k (in binary), how many k-tree-models does φ have?

Counting Tree-Models with Binary Bounds

Theorem

The following problem is $\#_d$EXPSPACE-hard and in $\#_d$2EXPTIME: Given an LTL formula φ and a bound k (in binary), how many k-tree-models does φ have?

- **Lower bound:**



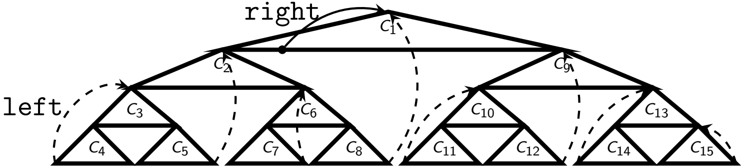
  - each inner tree has exponentially many leaves
  - tree has exponential height (thus, doubly-exponentially many inner trees)

Counting Tree-Models with Binary Bounds

Theorem

*The following problem is $\#_d$*EXPSPACE*-hard and in $\#_d$*2EXPTIME*:*
Given an LTL formula φ and a bound k (in binary), how many k-tree-models does φ have?

- **Lower bound:**



  - each inner tree has exponentially many leaves
  - tree has exponential height (thus, doubly-exponentially many inner trees)
- **Upper bound:** Guess tree of height k and model-check it

Conclusion

Overview of results:

	unary	binary
words	#P-compl.	$\#_d$PSPACE-compl.
trees	$\#_d$EXPTIME-compl.	$\#_d$EXPSPACE-hard/$\#_d$2EXPTIME

Conclusion

Overview of results:

	unary	binary
words	$\#\mathrm{P}$-compl.	$\#_d\textsc{Pspace}$-compl.
trees	$\#_d\textsc{Exptime}$-compl.	$\#_d\textsc{Expspace}$-hard/$\#_d 2\textsc{Exptime}$

Lower bounds: safety LTL, upper bounds: full LTL

Conclusion

Overview of results:

	unary	binary
words	#P-compl.	$\#_d$PSPACE-compl.
trees	$\#_d$EXPTIME-compl.	$\#_d$EXPSPACE-hard/$\#_d$2EXPTIME

Lower bounds: safety LTL, upper bounds: full LTL

Open problems:

- Close the gap!

Conclusion

Overview of results:

	unary	binary
words	#P-compl.	$\#_d$PSPACE-compl.
trees	$\#_d$EXPTIME-compl.	$\#_d$EXPSPACE-hard/$\#_d$2EXPTIME

Lower bounds: safety LTL, upper bounds: full LTL

Open problems:

- Close the gap!
  - Lowering the upper bound: how to guess and model-check doubly-exponentially sized trees in exponential space?

Conclusion

Overview of results:

	unary	binary
words	$\#$P-compl.	$\#_d$PSPACE-compl.
trees	$\#_d$EXPTIME-compl.	$\#_d$EXPSPACE-hard/$\#_d$2EXPTIME

Lower bounds: safety LTL, upper bounds: full LTL

Open problems:

- Close the gap!
  - Lowering the upper bound: how to guess and model-check doubly-exponentially sized trees in exponential space?
  - Raising the lower bound: how to encode doubly-exponentially sized configurations using polynomially sized formulas? Do games help?